AMENDMENTS TO THE CLAIMS

1. (original) In a computer having object-execution system code, a method of managing execution of software components in an object execution environment, the method comprising:

supporting an object execution environment from a system-provided run-time executive; and

responsive to a request from an immediate client to the run-time executive to create a software component instance in the object execution environment,

creating a component context object having stored therein a set of properties intrinsic to the requested software component instance and relating to managing execution of the requested software component instance within the object execution environment by system-provided code;

supplying to the immediate client a reference to the requested software component instance;

maintaining by the run-time executive an implicit association of the component context object to the requested software component instance.

2. (original) The method of claim 1 further comprising maintaining the component context object in existence and continuing the implicit association of the component context object to the requested software component instance during a lifetime of the requested software component instance.

3. (original) The method of claim 1 further comprising maintaining the component context object in existence and continuing the implicit association of the component context object to the requested software component instance until a final release of the requested software component instance.

4. (original) The method of claim 1 wherein the properties are set at creation of the component context object and maintained immutable for the lifetime of the requested software component instance.

5. (original) The method of claim 1 wherein the software component instance is one of a chain of software components whose creation was initiated by a base client that runs outside the object execution environment, and the set of properties comprises a client identifier indicative of the base client.

6. (original) The method of claim 1 wherein the software component instance is one of a set of ancestor/descendent-related software components, each of which being created in turn at request of its respective immediate ancestor software component of the set commencing with an original ancestor software component originally created at request of a base client, and wherein the set of properties comprises a client identifier indicative of the base client.

7. (original) The method of claim 6 wherein the client identifier indicates a base client resident outside of the object execution environment.

8. (original) The method of claim 6 wherein the set of related software components all are engaged in a collective task for the base client, and wherein the set of properties comprises an activity identifier indicative of the set of related software components.

9. (original) The method of claim 8 further comprising managing concurrency of execution within software components in the object execution environment that have the activity identifier stored as an intrinsic property in their respective context object.

10. (original) The method of claim 8 wherein the set of properties comprises a transaction reference indicative of a transaction encompassing data processing work of at least some of the related software components including the requested software component instance.

11. (original) The method of claim 6 wherein the set of properties comprises a transaction reference indicative of a transaction encompassing data processing work of at least some of the related software components including the requested software component instance.

12. (original) The method of claim 11 further comprising managing data processing work of software components in the object execution environment that have the transaction reference stored as an intrinsic property in their respective component context object on a transactional basis.

13. (original) The method of claim 6 wherein a second component context object is associated with the immediate ancestor software component for the requested software component instance, the method further comprising:

setting the properties in the component context object of the requested software component instance based on those in the second component context object.

14. (original) The method of claim 13 further comprising inheriting the properties in the component context object of the requested software component instance from those in the second component context object.

15. (original) The method of claim 13 further comprising setting at least some of the properties in the component context object of the requested software component instance to match those in the second component context object.

16. (original) The method of claim 13 wherein each software component of the set of ancestor/descendent-related software components has a respective component context object whose properties are inherited based on those of a third component context object implicitly associated to the original ancestor software component, and wherein the set of properties of the third component context object comprises a transaction reference indicative of a transaction encompassing data processing work of the set of ancestor/descendent-related software components, the method further comprising:

exposing an interface of the third component context object to the original ancestor software component, the component context object interface having a set complete indication method member and a set abort indication method member for invocation by the original ancestor software component, where the invocation of the set complete indication method member by the original ancestor software component is indicative of the set's data processing work being in a complete condition in which the transaction can be committed, and where the

invocation of the set abort method member by the original ancestor software component is indicative of the set's data processing work being in a complete condition in which the transaction cannot properly be committed;

invoking by the requested software component instance specific an appropriate one of the set complete indication or set abort indication method member of the third component context object interface prior to return from data processing work in the transaction that indicates the data processing work's condition;

where the set complete indication method member is invoked by the original ancestor software component prior to return from the data processing work, automatically attempting committal of the transaction without explicit programming instruction of the immediate client; and

where the set abort indication method member is invoked by the original ancestor software component prior to return from the data processing work, automatically aborting the transaction without explicit programming instruction of the immediate client.

17. (original) The method of claim 16 further comprising, responsive to the invocation of the appropriate one method member by the original ancestor software component prior to return from the data processing work, deactivating the original ancestor software component.

18. (original) The method of claim 1 wherein the set of properties comprises an activity identifier indicative of a set of software components inclusive of the requested software component instance all engaged in a collective task for a base client.

19. (original) The method of claim 18 further comprising managing concurrency of execution within software components in the object execution environment that have the activity identifier stored as an intrinsic property in their respective component context object.

20. (original) The method of claim 1 wherein the set of properties comprises a transaction reference indicative of a transaction encompassing data processing work of the requested software component instance.

21. (original) The method of claim 20 further comprising managing data processing work of software components in the object execution environment that have the transaction reference stored as an intrinsic property in their respective component context object on a transactional basis.

22. (original) The method of claim 20 further comprising:

exposing an interface of the component context object to the requested software component instance, the component context object interface having a plurality of method members for invocation by the requested software component instance to indicate a condition of the data processing work by the requested software component instance in the transaction; and

invoking by the requested software component instance specific of the method members of the component context object interface prior to return from data processing work in the transaction to indicate the data processing work's condition.

23. (original) The method of claim 22 wherein the method members of the component context object comprise a set complete indication method member, the invocation of which by the requested software component instance is indicative of the requested software component instance's data processing work being in a complete condition in which the transaction can be committed.

24. (original) The method of claim 23 further comprising, responsive to invocation of the set complete indication method member by the requested software component instance prior to return from the data processing work, deactivating the requested software component instance.

25. (original) The method of claim 23 further comprising, responsive to invocation of the set complete indication method member by the requested software component instance prior to return from the data processing work, automatically attempting committal of the transaction without explicit programming instruction of the immediate client.

26. (original) The method of claim 22 wherein the method members of the component context object comprise a set abort indication method member, the invocation of which by the requested software component instance is indicative of the requested software

component instance's data processing work being in a complete condition in which the transaction cannot properly be committed.

27. (original) The method of claim 26 further comprising, responsive to invocation of the set abort indication method member by the requested software component instance prior to return from the data processing work, deactivating the requested software component instance.

28. (original) The method of claim 26 further comprising, responsive to invocation of the set abort indication method member by the requested software component instance prior to return from the data processing work, automatically aborting the transaction without explicit programming instruction of the immediate client.

29. (original) The method of claim 22 wherein the method members of the component context object comprise a commit enabling indication method member, the invocation of which by the requested software component instance is indicative of the requested software component instance's data processing work being in a possibly incomplete condition in which the transaction can still properly be committed.

30. (original) The method of claim 29 further comprising, upon an attempt to commit the transaction at a time after the requested software component instance invokes the commit enabling indication method member, allowing the transaction to commit unless a condition of processing work by other than the requested software component instance causes an abort of the transaction.

31. (original) The method of claim 22 wherein the method members of the component context object comprise a commit disabling indication method member, the invocation of which by the requested software component instance is indicative of the requested software component instance's data processing work being in a possibly incomplete condition in which the transaction cannot properly be committed.

32. (original) The method of claim 31 further comprising, upon an attempt to commit the transaction at a time after the requested software component instance invokes the commit disabling indication method member, causing the transaction to abort.

33. (original) The method of claim 1 wherein a second component context object is associated with the immediate client executing in the object execution environment, the method further comprising:

setting the properties in the component context object of the requested software component instance based on those in the second component context object.

34. (original) The method of claim 33 further comprising inheriting the properties in the component context object of the requested software component instance from those in the second component context object.

35. (original) The method of claim 33 further comprising setting at least some of the properties in the component context object of the requested software component instance to match those in the second component context object.

36. (original) The method of claim 33 further comprising issuing by the immediate client a method invocation to a create instance method member of an interface supported on the second component context object to thereby request creation of the requested software component instance.

37. (original) The method of claim 1 further comprising, on request of the software component instance to a get context programming interface of the run-time executive, providing a reference for use in accessing the component context object to the software component instance.

38. (original) The method of claim 1 further comprising, on request of any of the software components in the object execution environment to a get context programming interface of the run-time executive, providing a reference for use in accessing said any software component's respective component context object.

39. (original) The method of claim 1 wherein the computer further has a security facility requiring a user to log-on under one of a plurality of user identities configured on the respective computer system, at least some of the software components in the object execution environment form at least part of a component-based application program having defined therein a set of abstract user classes and access privileges of the abstract user classes, the computer having an administrative configuration associating the user identities to the abstract user classes and having an access control for enforcing access limitation of a user whose user identity is associated in any of the abstract user classes to the access privileges of the associated abstract user class, the method further comprising:

exposing an interface of the component context object to the requested software component instance, the component context object interface having a method member for programmatically querying whether a caller is executing on behalf of a user associated in a specified abstract user class; and

in response to invocation of the method member by the requested software component instance which invocation specifies a particular abstract user class, returning an indication of whether a caller of the requested software component instance executes on behalf of a user associated in the particular abstract user class.

40. (original) The method of claim 39 wherein the access control can be selectively enabled to enforce the access limitation with respect to any software component of the object execution environment, the exposed interface of the component context object also having a further method member for querying whether the access control is enabled with respect to the requested software component instance, the method further comprising:

in response to invocation of the further method member by the requested software component instance, returning an indication of whether the access control is enabled with respect to the requested software component instance.

41. (original) In a computer, a software component execution runtime executive for providing services to support execution of software components in an object execution environment, the runtime executive comprising:

program code supplying an implementation of component context objects for accompanying individual instances of the software components executing in the object execution environment, the component context objects having stored therein a set of context properties intrinsic to their respective accompanied individual software component instance and relating to managing execution of the accompanied individual software component instance within the object execution environment;

program code operating responsive to a request of an immediate client for creation of an instance of a request-specified software component to cause creation of a component context object to accompany the requested software component instance and to return a requested software component instance reference back to the immediate client; and

program code for maintaining an implicit association of the created component context object with the requested software component instance throughout a lifetime of the requested software component instance.

42. (original) The software component execution runtime executive of claim 41 wherein, once set at creation of the component context object, at least some of the context properties stored in the component context object remain immutable for the lifetime of the requested software component instance.

43. (original) The software component execution runtime executive of claim 41 further comprising:

program code of a programming interface to the software component execution runtime executive, the programming interface operating responsive to a request of the requested software component instance to return an interface pointer of an interface of the component context object.

44. (original) The software component execution runtime executive of claim 41 further comprising:

program code implementing an interface of the component context object exposed to the requested software component instance, the interface having a component creation method member for invocation by the requested software component instance to cause creation of a descendant software component instance in the object execution environment; and

program code operating responsive to the invocation of the component creation method member by the requested software component instance to create a second component context object to accompany the descendant software component instance and having context properties stored in the second component context object that are inherited from the context properties of the component context object accompanying the requested software component instance.

45. (original) The software component execution runtime executive of claim 41 further comprising:

program code implementing an interface of the component context object exposed to the requested software component instance, the interface having a transaction completion indication method member for invocation by the requested software component instance prior to return from data processing work for the immediate client to provide an indication to the software component execution runtime executive that the data processing work has been successfully completed.

46. (original) The software component execution runtime executive of claim 45 further comprising:

program code operating responsive to the invocation of the transaction completion indication method member by the requested software component instance to deactivate the requested software component instance upon the requested software component instance's return from the data processing work.

47. (original) The software component execution runtime executive of claim 45 further comprising:

program code operating responsive to the invocation of the transaction completion indication method member by the requested software component instance to initiate a commit phase of transactional processing involving the data processing work of the requested software component instance.

48. (original) The software component execution runtime executive of claim 41 further comprising:

program code implementing an interface of the component context object exposed to the requested software component instance, the interface having a transaction abort indication method member for invocation by the requested software component instance prior to return from data processing work for the immediate client to provide an indication to the software component execution runtime executive that the data processing work has been unsuccessfully completed.

49. (original) The software component execution runtime executive of claim 48 further comprising:

program code operating responsive to the invocation of the transaction abort indication method member by the requested software component instance to deactivate the requested software component instance upon the requested software component instance's return from the data processing work.

50. (original) The software component execution runtime executive of claim 48 further comprising:

program code operating responsive to the invocation of the transaction abort indication method member by the requested software component instance to initiate an abort phase of transactional processing involving the data processing work of the requested software component instance.

51. (original) The software component execution runtime executive of claim 41 further comprising:

program code implementing an interface of the component context object exposed to the requested software component instance, the interface having a transaction disable indication method member for invocation by the requested software component instance prior to return from data processing work for the immediate client to provide an indication to the software component execution runtime executive that the data processing work should not be committed pending further processing.

52. (original) The software component execution runtime executive of claim 51 further comprising:

program code operating responsive to the invocation of the transaction disable indication method member by the requested software component instance to disable transactional processing involving the data processing work of the requested software component instance from committal.

53. (original) The software component execution runtime executive of claim 51 wherein the interface further has a transaction enable indication method member for invocation by the requested software component instance prior to return from data processing work for the immediate client to provide an indication to the software component execution runtime executive that the data processing work can be committed, the software component execution runtime executive further comprising:

program code operating responsive to the invocation of the transaction enable indication method member by the requested software component instance to again enable transactional processing involving the data processing work of the requested software component instance to commit.

54. (original) The software component execution runtime executive of claim 41 further comprising:

program code implementing an interface of the component context object exposed to the requested software component instance, the interface having a query transaction participation method member for invocation by the requested software component instance; and

program code operating responsive to the invocation of the query transaction participation method member by the requested software component instance to provide information back to the requested software component instance whether data processing work of the requested software component instance forms part of a transaction.

55. (original) The software component execution runtime executive of claim 41 further comprising:

program code implementing an interface of the component context object exposed to the requested software component instance, the interface having a query role membership method member for invocation by the requested software component instance to programmatically control access; and

program code operating responsive to the invocation of the query role membership method member by the requested software component instance to provide information back to the requested software component instance indicative of whether a user on whose behalf the requested software component instance is executing is associated in a specified role, wherein the role is an abstract class of users defined for a component-based application program and associated to user identities employed in a security facility of the computer.

56. (original) The software component execution runtime executive of claim 41 wherein the context properties stored in the component context object comprise a client identifier indicative of a base client that initiated creation of a set of software components in the object execution environment inclusive of the requested software component instance to perform a data processing activity.

57. (original) The software component execution runtime executive of claim 41 wherein the context properties stored in the component context object comprise an activity identifier indicative of a set of software components in the object execution environment inclusive of the requested software component instance all engaged in a single collective data processing activity.

58. (original) The software component execution runtime executive of claim 57 further comprising program code operating to control concurrency of execution in the set of software components.

59. (original) The software component execution runtime executive of claim 57 further comprising program code operating to limit concurrent execution in the set of software components to a single logical thread of execution.

60. (original) The software component execution runtime executive of claim 41 wherein the context properties stored in the component context object comprise a transaction reference indicative of a transaction encompassing data processing work of a collection of participating software components in the object execution environment inclusive of the requested software component instance.

61. (original) The software component execution runtime executive of claim 60 further comprising program code operating to transactionally manage the data processing work of the collection of participating software components.

Claims 62-68. (canceled)